EXHIBIT 119

Automatic Whitelisting for AWBid

status: up-to-date, **launched on 12/5/2013**
reviewed on 9/23/2013

joanyuan@google.com
with valuable inputs from dimakuzmin@, eujin@, myl@, dannyk@, sduan@, mingtianni@, myzhong@, vivekrao@, oren@ and others

go/awbid-whitelist

Commented [1]: Longer term we should integrate with spam filters using RTSF server in GDN.

GDN is already working on this, if not we will do this as part of skyray.

In the meanwhile I am also doing this for Xbid see go/xbid-spam, however i was planning to do this in the mixer, since we need geo signals.

We can talk about whether we can reuse that server for your case also.

Commented [2]: it appears this is a slight different use case than go/xbid-spam. We want to do it at the XBFE level, with a limited set of featuers (from GenericBidRequest), before anything is logged; so that we will not pass questionable traffic to the backend at all (this approach will also reduce spam input to xbid). I think it may be a good idea to integrate with RTSF later; if we can provide all the signals they need.

Commented [3]: Yeah, right now we are basically going for an automated/cautious version of current awbid blacklist. This is pre-bid filtering, since gdn already gives us post-bid spam detection. We definitely should keep RTSF integration in mind. Remains to be seen if enough signals can be made available during bid time, current numbers for awbid query-level spam detection are much smaller than the click/view spam rates where we get more signals.

Commented [4]: Right, the signals are much weaker because they're coming in through bid requests rather than being constructed by show_ads.js. Also, the filters applied in real-time will always be a strict subset of the full available list.

Objective

A dynamic, automated way of ramping up traffic for new domains when they send queries to AWBid, and ramp down their traffic if they prove to be too spammy. The goal is to have an efficient, reliable, fully automated way of throttling traffic, replacing the manual blacklist process we currently have.

There are two goals for this design

1. provide a scalable and safe way of ramping up new publishers, instead of doing it manually. This design will allow brand new, non spammy publishers to ramp up to 100% traffic within a week.
2. protect GDN from paying out more than what we charge the advertisers by limiting spammy traffic into GDN.

These are the non goals of the design

1. Increasing Google profit by increasing revenue/payout >= 1. This will be considered in the context of bidding strategies.

Background

AWBid is a system that allows AdWords customers to bid on non AdSense or ADX publishers. In this case, we serve traffic from third party exchanges, many of which don't have the same rigorous spam detection mechanism in place as Google does. Since AWBid beta went live in May 2013, we have observed a higher percentage of impression and click spam rate (compared with adsense and adx traffic). A handful of publishers account for majority of the spams. Bidding on spam traffic wastes resource, decreases available bandwidth, and skews training data feeding into our machine learning system.

	spam impression rate	spam click rate
AWBid	4% - 30%	10% - 70%
ADX		7% - 8%
AFC online	~1.2%	0.07% - 0.2%

Commented [5]: If our pCTR model is accurate, spam should have no effect on profit as long as we only pay for clean impressions. Ramping pubs down will therefore reduce profit. This can be traded-off against the value of resource savings (some monetary value) to estimate extent of ramping down. The net effect will be to maximize profit + resource saved, while also achieving reduced spam rates.

Commented [6]: If we don't pay for spam, why is this relevant?

Commented [7]: This is a related goal, if your objective is to protect GDN payout out too much.

I'm not clear on the exact problem with spam from reading this introduction. Is it that:

a) Spam is a waste of resources - we serve the ad and later don't pay.

b) Spam makes the pCTR inaccurate and therefore we overbid

c) We should not "reward" spammy pubs by sending them traffic.

Other...?

Commented [8]: Shouldn't we explicitly state this as a non goal and aim for neutrality for both spam filtering and pctr calibration and consider any arbitrage only in other contexts like bidding strategies?

Commented [9]: noted.

Commented [10]: Add numbers?

Commented [11]: added adx numbers, still trying to track down afc numbers

Commented [12]: Is this true if spam is filtered before training models?

Commented [13]: Yes, as far as the AdX pCTR calibration model is concerned

Commented [14]: Is this overall before any of the blacklisting was applied?

Commented [15]: 70% end was before the blacklisting; 10% or so is nowadays

Currently, all new publishers associated with the approved exchange can send traffic to AWBid, and we employ a manual process to block publishers that have high spam rate. Domains whose spam rate is above a certain threshold are flagged by spam team, and engineers manually enter them into a blacklist. XBFE picks up the blacklist and filters out those domains/publishers at serving time. This process does not scale well and the blacklist needs to be periodically reviewed to make sure that it's still accurate, as legitimate domains can fall victim to spammy traffic due to bad implementations, attacks, etc.

In this design doc, we propose an automated process that can ramp up/down traffic based on the daily spam rate.

Overview

We use a whitelist approach to decide which traffic will be relayed to BOW (content ads front end) -> AWBid. The key of the whitelist is domain/sub_web_property, and each entry corresponds to a floating value between [0,1], indicating of probability of this request will be sent to AWBid. It's a pseudo whitelist approach, as we do let in a low percentage of new domain traffic, to account for new publishers; since we do not get enough signals (full url, geo, etc) at query time, since those requests came to us from third party exchanges, and they filter out signals not relevant to cookie targeting (or due to unwillingness to share too detailed data with Google; or simply because of feature disparity).

AWBid dashboard already runs a daily program to output publisher domain stats, including clean impressions, all impressions; clean clicks and all clicks -- among other stats. We will calculate clean impression rate and clean click rate based on that. A new offline program will calculate the traffic throttle threshold for each sub_web_property entry each day, based on the spam stats.

This whitelist is loaded by XBFE at serving time, and used to determine whether it should pass a specific request to awbid, based on a pseudo random number generator and the threshold corresponding to the sub_web_property entry in the whitelist. If such an entry is not located, indicating it's a new sub web property, XBFE lets in the traffic based on a pre-defined probability (e.g. 10%).

Commented [16]: Are only looking at awbid traffic to generate the data? Or can we also learn something from xbid or even adx logs?

Commented [17]: Related - can we get a list of all pubs / sub_wed_property that was blacklisted from adx?

Commented [18]: current plan is only awbid data. I think we can look at xbid/adx logs --- I need to check if we have all the stats available from those logs and get permission to use those logs. regarding blacklisted adx pubs, I will check with spam ops team , not sure what format those list will be (last time I asked, for adsense publishers, it's contact info, phone numbers, etc)

Commented [19]: You probably should explain this a little more. We need to let in some percentage of traffic through since we don't get enough signals at query time due to the fact that awbid traffic comes in as server-to-server calls. May be list query/view/click spam rates for awbid for comparison.

Commented [20]: updated

Commented [21]: Or simply because it's not something they have any reason to implement in their own tags and bid request protocol (i.e. it's as much a case of lack of feature parity in their js tags as it is deliberate filtering).

Commented [22]: updated.

Infrastructure(s)

There are two parts to this design

1) offline program that generates the whitelist daily and push it out to production. Since Xbid has already set up supermessenger channels and an MLT instance, we will add to

Commented [23]: one simple approach here is to use the in-memory MLT instead of a external server.

How much data are we talking about?

Commented [24]: as pointed out in the doc, anywhere between 5-50 meg, most likely 20-30 in the foreseeable future

existing infrastructure for XBFE.

here's the list of jobs, at a high level

- a job to generate the whitelist and save it in sstable.
- a producer program that reads from whitelist sstable and writes to supermessenger channel (a set of CNS files). We could consider writing the hash value of the key instead of the string itself. This will take less space, also enables faster lookup.
- file a ticket for SRE to 'turn on' that channel (essentially ads copy service)
- modify XBid mlt config so that it loads the new channel

2) serving time change. We will ~~need XBFE to maintain a stubby connection to MLT server.~~ have XBFE load a local MLT instance in memory, since the file size is small. XBFE currently does not do this; but Xbid mixer does.

Commented [25]: Will you run in multiple locations and write to different "colours" of the channel to make it resilient to pcrs?

Commented [26]: yes. it will follow the setup of existing xbid supermessgener channels.

Commented [27]: Does it need to be a separate from the previous job, or write to an intermediate sstable?

Commented [28]: the producer job could be the same job. we want to keep an sstable on disk for debugging /sanity check.

Commented [29]: Why not just load the mlt in the xbfe since it's small?

Commented [30]: XBFE/SRE advised against the approach of loading into each xbfe task memory; esp when the list has potential to grow. They want to get rid of the existing blacklist loaded in memory too (don't want it loaded in memory)

Detailed Design

Whitelist Format

We will start at the sub web property level. The whitelist contains list of key/value pair, where each key is sub_web_property, and the value is a decimal number between 0.1 and 1.0, indicating the percentage of traffic that AWBid wants to bid on for this particular sub_web_property. Both web property(exchange id) and sub web property will be extracted from GenericBidderRequest, they are in numeric format.

Commented [31]: It's probably enough to use the shorter exchange id.

Commented [32]: updated

Later, as we get enough data at the domain level, we may add domain level threshold entries.

Alternative levels of entries were considered, refer to the section 'Alternative Designs Considered'.

~~The whitelist contains list of key/value pair, where each key is of the format ***domain:sub_web_property***, and the value is a decimal number between 0.1 and 1.0, indicating the percentage of traffic that AWBid wants to bid on for this particular domain/sub_web_property combination.~~

~~A simple domain to traffic threshold mapping is not sufficient because we have observed some domains (especially large domains) come in via different sub web properties, sometimes even different web properties (ad exchanges). For example, on 9/3/2013, the domain 'bbc.com' was associated with 15 sub web properties, with clean impression rate~~

~~ranging anywhere from 3% to 100%. (refer to 'alternative designs considered' section).~~

1. ~~A note on the size of the whitelist -- a quick dremel query shows that on any given day, there exists anywhere from 7000 to 10,000 pairs of <domain:sub_web_property> from the logs currently (09/2013). If we assume 512 byte for the key length -- top level domain should be much shorter than that, and sub web property is "bidder-xxxxxx' where xxxx is a number, rarely even 10 digits long -- our current list should be less than 5 Meg in size. Factor future growth as we sign up more exchanges, multiple that number by 100, we'd be safely under 1G in the extreme case. Typically much less than 50 Meg in the foreseeable future. To further save space, we could save hash value for the keys to 64 or 128 bits.~~

Whitelist Generation

Actual rampup plan discussion is outlined in a separate doc.

We already have an offline daily sawzall job that monitors all domains we get traffic from, along with impressions/clicks they get from AWBid. The sawzall outputs contains structure AWBidStats, which is defined here.

Here are some definitions of the terms:

- served_impressions (aka eligible impressions): impressions served from cat2. Note those are NOT the same as impressions shown on the ad exchanges eventually, as these impressions will need to compete in exchanges auction in order to be shown.
- all_impressions (aka transacted impressions): impressions that were rendered on the ad exchange and were viewed.
- clean_impressions (aka revenue impressions): impressions that are not spammy.
- all_clicks: all clicks (including spammy clicks)
- clean_clicks (aka revenue clicks): all non spammy clicks.

We will add another offline mapreduce/flume job (whitelist writer) that outputs a new whitelist (in sstable format) to be used in production serving. The job will emit previously seen sub_web_property entries, and their corresponding traffic threshold levels. The input to the flume job are csv milldump of the publisher dashboard sawzall job and the current serving whitelist (sstable). The output of the flume job is the new whitelist (sstable).

Based on the milldump csv file from the publisher dashboard job, for each sub_web_property value, the whitelist writer job calculates the clean impression rate

Commented [33]: is it bad to treat it as a single domain though?

I suppose we might penalize an entire domain if just one or two web properties are bad. What if we don't have enough data for all the subwebproperties, could it be better to just group it by domain? Although that could be just part of the offline job since we could just write the same ratio for all the sub web properties for the same domain.

On the other hand, if we do domain, we'd be able to ramp up data across exchanges, which is a good thing.

Commented [34]: I don't think it's a good idea to treat it as a single domain, as you pointed out, that would penalize the whole domain if a sub wp is bad. Also, are you sure ramping up data across exchange is a good idea? there could be (new) bad sub wp from one exchange that could fluctuate traffic for that domain f

Commented [35]: I think that might be ok, we should discuss this more with the other people in the team. I

Commented [36]: Why exclude any of these scenarios? We could give ourselves the flexibility to

Commented [37]: that makes sense -- having multiple level of 'rules'. We should be cautious though, this

Commented [38]: This sounds like what the original remarketing hierarchical model was meant to do. e.g.

Commented [39]: Ok, how about just two levels? swp and swp:domain. It seems it makes sense

Commented [40]: per our discussion, we will start with entries at sub web property level. yes, mlt does suppo

Commented [41]: count distinct domain from adquery in general gives me about 18M domains, but I guess

Commented [42]: right, the number cited is from awbid queries.

Commented [43]: This is small enough that we can load directly

Commented [44]: 5 Meg is the current size (2 exchanges), we anticipate the upper bound to be 30-

Commented [45]: If 50 meg is a reasonable limit, I think loading it in xbfe itself still makes sense. That

Commented [46]: I don't mind doing that, except that XBid/XBFE side (especailly SRE) strongly prefers us

Commented [47]: We should probably have an option to merge in a manually created whitelist, that would

Commented [48]: supermessenger channel takes care of that automatically. We can configure to canary our

Commented [49]: I thought the question was more about a longer term whitelist that would be applied to

Commented [50]: this would be fairly easy to do from the producer job, to 'pre-populate' the resulting whitel

(clean_impressions/ transacted_impressions) and clean click rate (clean_clicks/ all_clicks) for the past day. ***Based on those rates, and previous day's whitelist***, new traffic threshold is calculated as

1. if an entry has no clean impressions and no transacted impressions, indicating no views has taken place, increase previous day's threshold until 100%.

2. if an entry has clean_impression_rate >= 20%, but not any clicks (all_clicks = 0), increase previous day's threshold until 100%.

3. if an entry has clean_impression_rate <= 20%, but not any clicks (all_clicks = 0), increase previous day's threshold until at 80%.

4. if an entry (sub_web_property) has both clean impression and clean clicks,
    a. if its clean impression rate and clean click rate is above certain numbers x, y (where x and y are GDN level clean impression rate and clean click rate, plus some delta to account for deviation), then threshold is set to 1 -- meaning if their spam pattern is on par with GDN / ADX level spam rate, then do not throttle.
    b. its threshold is calculated as a moving average of current clean_impression_rate and clean_click_rate. here's the formula --

    new_threshold = a * old_threshold + (1-a) * calculated_threshold

    *where calculated_threshold = (clean_impression_rate * clean_click_rate);*

    here
    clean_impression_rate = (clean_impressions + historical_awbid_clean_impression_rate * N) / (transacted_impressions + N);

    clean_click_rate = (clean_clicks + historical_awbid_clean_click_rate * M) / (all_clicks + M).

    If the final number < 10%, drop this entry from whitelist (so that we will apply the min threshold -- 10% -- at serving time).

the above rate calculation is used to avoid rate being magnified when # of click/impressions are small.

When "increasing" traffic threshold level, we follow a tiered threshold defined below.

10% -> 20% -> 40% -> 60% -> 80% -> 100%

Note that threshold decrease is achieved by multiplying clean impression rate and clean click rate, so that when either one is too small, traffic will automatically be throttled down.

Commented [51]: There should probably be provision for long term blacklisting entries that oscillate over longer periods.

Commented [52]: can you add a definition of clean and transacted impressions above?

Commented [53]: done

Commented [54]: Can you explain the difference between 2 and 3 a bit more? 3 seems to be a case with impression spam > 80%. That seems pretty bad. :) Why should such entries still ramp up to 80%? Also, how did you pick 20% and 80%?

Commented [55]: 20% and 80% are just picked randomly. :) --- coming up with an efficient rampup plan is something we are going to tweak/experiment with. this is a very simple, naiive first version, to get things going.

difference between 2) and 3) are just in 3), we have non 0 clean click rate --- the guideline in this version is : keep ramping up until we get some clicks; but if clean impression rate is too low, ramp up only to 80% of traffic.

Commented [56]: What's the thinking behind this formula? I am not sure what this gets us. You are roughly saying that we should get traffic through in the proportion equal to its clean traffic rate. Like Eujin points out below, this means that all publishers will be ...

Commented [57]: 1) the thinking behind this formula is to come up a simple, first version way of throttle traffic based on their spam rate. I agree with Eujin and you ...

Commented [58]: there probably should be a corresponding threshold where if we compute that the spam rate for a web property is within some margine ...

Commented [59]: that's a great point. I am trying to see if I can dig up a number from either spam dashboard or somewhere, as this number probably ...

Commented [60]: overall comments:
1. what do we do with a totally new pub - how do we limit damage from the first day; also just looking at the ...

Commented [61]: 1) we only let in 10% of traffic from a pub we've never seen, since we are doing at the sub web property level now (not the domain level), so tha ...

Commented [62]: Just to make sure I understand this correctly - suppose an entry generates only spammy clicks and has very low CTR in general, then the ...

Commented [63]: your understanding is correct. and your example is definitely possible, if you run this dremel query: ...

Commented [64]: Yes, I meant clean clicks.
And agreed, we can definitely live with this redundant space - just wanted to make sure that I understand th ...

Commented [65]: thank you for your comment!! really appreciate it. no need to be apologetic! :)

We drop any entry that has threshold level less than 10% to save space, as 10% is automatically applied for all entries not on the white list.

We can run experiments with different ramp up plan to measure the impact. The metric we want to optimize is profit.
The offline flume program will need to consult current serving threshold sstable to determine the current throttle level of the sub_web_property key, and increase based on previous day's level.

Note this approach tends to penalize traffic with high click spam rate, e.g. consider two entries from logs from 9/4

key	clean impr.	all impr.	clean impr rate	clean clicks	all clicks	clean click rate	threshold
key1	51	65	**0.785**	2	7	**0.2857**	**22%**
key2	351	451	**0.778**	4	30	**0.1333**	**10%**

where bold numbers in the above table were calculated from other numbers. Note even though key2 has many more impressions and clicks, its higher click spam rate will cause it to have lower threshold the next day. Since we pay publishers by impressions and can only charge by clean clicks, this suggested approach will optimize Google's revenue.

Optionally, the whitelist writer job should also have the ability to generate the whitelist from a predefined sstable (similar to a bootstrap sstable) that has some predefined entries (e.g. allow 0% for certain sub wps and 100% for other sub wps). The writer will have the option to 'merge' the predefined sstable entries with newly calculated entries and write the result to the serving sstable. The point is that basing rampup plan completely on local information may not be sufficient, there may eventually be a need to take into account of information such as repeated oscillation over longer period of time.

Pushing Data to Production

Next, we will have a producer job that reads from the sstable and writes to supermessenger channels, so that the whitelist can be accessed by production jobs. Supermessenger channels are essentially a set of files on CFS. The producer job will utilize SuperMessengerWriter to update the channels once a day, more frequent if needed.

We will need to file a ticket (similar to this one) to set up a new supermessenger channel so

Commented [66]: Assuming current situation where we can get paid by our spam numbers, won't profit be maximized at zero throttle rate? We probably need to look at tradeoff between profit reduction and residual spam rate.

Commented [67]: agreed. also, we do not get paid for spam numbers, we just get away with not paying for spam impressions.

Commented [68]: Should we also pay attention to cases where the click and impression spam rates are detected as very different or is that sufficient captured by independent checks on both of them?

Commented [69]: I think it would be interesting to do some analysis on such examples, may be forward a few to spam team for their opinion.

Commented [70]: myl/dima, we (sduan@) will run a check on existing data -- will calculate clean click/impression rate per domain and see how much they vary, we will go from there.

Commented [71]: actually, I just ran a dremel query, and it looks like we do have entries with very different clean impression rate and clean click rate --- I will send out an email to show some sample data. at this point, it looks to me it could be accounted for as multiplication of the two , but I am open for other opinions

Commented [72]: Agree with Dima - profit would be optimized by letting bidding algorithms consider 100% of impressions (we don't pay for spam).

To really optimize, you have to (for example) assign a cost for processing an impression that turns out to be spam later. Then optimize for expected revenue - expected cost.

that we can push our white list to servers.

Serving Changes

Once pushed into supermessenger channel, the whitelist will then be loaded into an in memory MLT instance for XBFE, since data is small enough. (when data grows, we will consider moving it to a separate mlt server instance). ~~an MLT instance shared between XBFE and Xbid mixer. This MLT instance has been set up by the Xbid team, and we only need to modify its config to add the new channel.~~

Currently, XBFE does not interact with MLT server. So we need to modify XBFE so that it ~~maintains a stubby connection to MLT~~ loads a local MLT instance. It should probably be done in SplitBidFlow, in the init functions.

At serving time, XBFE queries MLT instance to determine whether to send a particular query to CAFE. This logic will be placed in cafe-control-flow.cc; or maybe split-bid-flow.cc. XBFE will also need to extract sub_web_property from repeated list site_id field in GenericBidRequest, so that it can construct the key entry to query the whitelist. ('domain' is a separate field in GenericBidRequest).

XBFE translate string sub_web_property into a hash value, then look up from the MLT interface using a lookup client such as this one or this one. If such entry is found in the whitelist, a float number is returned indicating what percent of traffic that should allowed through XBFE. If such entry is not found in the whitelist, a default threshold (10%) is applied. XBFE queries a pseudo random number generator for a number between 0 to 1000, and drops the request if it is greater than the (threshold * 1000) value. XBFE should keep a total counter of requests dropped due to throttling. Note this throttling only happens for CAFE, it does not affect Xbid serving path, XBFE will continue to pass this request to XBid.

Commented [73]: This will also be logged in Awbid logs, if I ever turn them on...

Commented [74]: when is it going to be turned on? :) -- What I meant here is that they'd be recorded in varz though, which will be helpful in addition to the logs.

Setting up Experiments

We will set up experiments to measure the impact on profit for different rampup plan XBid/XBFE does not use Mendel experiment framework like cat2 does. They do have a simple experiment framework to divert traffic into different groups.

Here's the change we need to make in order to set up experiments to measure profit impact for different ramp up plans

- cookie based experiments

xbfe does not support cookie based experiment, so we will need to simulate this by using a hash function to map cookie id to cookie mod, like this one. This mapping will be called after cookie parsing and sanity checking is done here. We will have to save the calculated cookie

mod into a newly added field in GenericBidRequest (say, 'cookie_mod') so that it can be used to divert experiment later on.

- custom configurations

custom configuration setting for each experiment is defined in BidRequestParams. We will add a string flag for 'rampup plan name', which corresponds to the supermessenger channel name for the ramp up plan (different plans will have different names). We will also add an int64 field indicating the corresponding cat2 experiment ids they map to.

- setting experiments

cookie experiment diversion logic is added in the experiment setting gcl file. We also set the corresponding BidRequestParam there. The criteria is 'awbid traffic only' and cookie mod ranges.

- translating xbid experiments to cat2 experiments

In split-bid-flow.cc, we retrieve BidRequestParam from FrontendRequest.data<ExperimentQueryState>; pass the name of the rampup plan and the cat2 experiment id to cafe-control-flow.cc. If we decide to pass the request to cafe, we will need to pass the cat2 experiment id. It will be set in function CafeRequestUtil::FillCafeRequest as a '***eid=***xxxx' parameter to CAFE.

note that CAFE only allows a whitelisted version of experiments from 'eid=' fields, so we need to make sure the cat2 experiment ids are added in the CAFE whitelist here.

To analyze experiment's impact on profit, we can get data from cat2 logs, using the cat2 experiment ids.

Commented [75]: Will you make XBFE listen to the channel? Or do you pass this string to cafe?

Commented [76]: right, xbfe will load a local mlt instance (not there yet, will be implemented) which has different rampup plan files. Each rampup file corresponds to a named channel. it will be a local lookup, not passing to CAFE.

Commented [77]: Why a string field?

Commented [78]: okay I stand corrected. experiment_id is defined as int64 in cat2 world, so we will do the same here.

Commented [79]: How do you express this criteria? Is there a field in GenericBidRequest?

Commented [80]: yes, awbid request comes from a certain predefined exchanges, so we'd be filtering based on GenericBidRequest.Exchange field (e.g. CASALE_MEDIA, etc)

Commented [81]: OK. Please keep the expression concise. The criteria will be evaluated for each bid request.

Commented [82]: ok, it will be of the format

lhs_field_name = 'GenericBidRequest.exchange'
comparator = 'EQ'
rhs_value = 'CASALE_MEDIA'

ORed with other 3-4 exchanges we serve.

Alternative designs considered

the follow alternative approaches were considered

Where should the throttle happen in the serving stack

We decided to do it in XBFE since that's the outer-most layer of the serving path. It makes sense to drop requests we don't want to serve there, saves further processing time/resource.

An alternative approach to drop the request within cat2 was also considered, that is further down the serving stack. Most importantly, doing it there would affects cat2 'coverage' (the percent of times where we return an ad for a request), so it's not desirable to do it within cat2.

Whitelist format

At domain/sub web property level

The whitelist contains list of key/value pair, where each key is of the format ***domain:sub_web_property***, and the value is a decimal number between 0.1 and 1.0, indicating the percentage of traffic that AWBid wants to bid on for this particular domain/sub_web_property combination.

A simple domain to traffic threshold mapping is not sufficient because we have observed some domains (especially large domains) come in via different sub web properties, sometimes even different web properties (ad exchanges). For example, on 9/3/2013, the domain 'bbc.com' was associated with 15 sub web properties, with clean impression rate ranging anywhere from 3% to 100%. (refer to 'alternative designs considered' section).

A note on the size of the whitelist -- a quick dremel query shows that on any given day, there exists anywhere from 7000 to 10,000 pairs of <domain:sub_web_property> from the logs currently (09/2013). If we assume 512 byte for the key length -- top level domain should be much shorter than that, and sub web property is "bidder-xxxxxx' where xxxx is a number, rarely even 10 digits long -- our current list should be less than 5 Meg in size. Factor future growth as we sign up more exchanges, multiple that number by 100, we'd be safely under 1G in the extreme case. Typically much less than 50 Meg in the foreseeable future. To further save space, we could save hash value for the keys to 64 or 128 bits.

At domain level

Initially, it was decided that the key of the whitelist will be domains. Upon closer examination, it appears that the same domain can be sent from different exchange/publishers, often with very different clean impression rate. For example, here's one query based on data from 09/05/2013,

select domain, clean_impressions, transacted_impressions, clean_impressions/transacted_impressions as imp_prec from awbid.pubside.20130905 where domain = "bbc.com" order by imp_prec desc;

domain	clean_impressions	transacted_impressions	imp_prec
bbc.com	5	5	1.0
bbc.com	11	11	1.0
bbc.com	27	27	1.0
bbc.com	304	304	1.0
bbc.com	45	45	1.0
bbc.com	117	117	1.0

Commented [83]: is it bad to treat it as a single domain though?

I suppose we might penalize an entire domain if just one or two web properties are bad. What if we don't have enough data for all the subwebproperties, could it be better to just group it by domain? Although that could be just part of the offline job since we could just write the same ratio for all the sub web properties for the same domain.

On the other hand, if we do domain, we'd be able to ramp up data across exchanges, which is a good thing.

Commented [84]: I don't think it's a good idea to treat it as a single domain, as you pointed out, that would penalize the whole domain if a sub wp is bad. Also, are you sure ramping up data across exchange is a good idea? there could be (new) bad sub wp from one exchange that could fluctuate traffic for that domain fo ...

Commented [85]: I think that might be ok, we should discuss this more with the other people in the team. I suspect that it's beneficial to use domain to share ...

Commented [86]: Why exclude any of these scenarios? We could give ourselves the flexibility to whitelist on any of the levels - domain only, swp only ...

Commented [87]: that makes sense -- having multiple level of 'rules'. We should be cautious though, this introduces another level of complexity (which rule to ...

Commented [88]: This sounds like what the original remarketing hierarchical model was meant to do. e.g. If we don't find the sub webproperty, then we try to use ...

Commented [89]: Ok, how about just two levels? swp and swp:domain. It seems it makes sense to white/blacklist on swp level too. Btw, does mlt ...

Commented [90]: per our discussion, we will start with entries at sub web property level. yes, mlt does support multi lookup

Commented [91]: count distinct domain from adquery in general gives me about 18M domains, but I guess its unlikely we will get that big. :)

Commented [92]: right, the number cited is from awbid queries.

Commented [93]: This is small enough that we can load directly

Commented [94]: 5 Meg is the current size (2 exchanges), we anticipate the upper bound to be 30-50 meg bytes in the foreseeable future (10 exchanges), ...

Commented [95]: If 50 meg is a reasonable limit, I think loading it in xbfe itself still makes sense. That would be like 1.5% of xbfe ram reservation.

Commented [96]: I don't mind doing that, except that XBid/XBFE side (especailly SRE) strongly prefers us using MLT -- allows all data in one centralized place; ...

bbc.com	134	134	1.0
bbc.com	380	381	0.997375328084
bbc.com	2023	2042	0.99069539667
bbc.com	395	399	0.989974937343
bbc.com	710	719	0.987482614743
bbc.com	535	543	0.985267034991
bbc.com	21	22	0.954545454545
bbc.com	10	272	0.0367647058824
bbc.com	0	0	NULL

domain bbc.com has clean impression rate ranging from 3% to 100% over the network, depending on which exchange/publisher it is coming from.

Ideally, we'd use <domain, sub_web_property, web_property> as key into the whitelist. But in reality, sub_web_property level usually suffice -- so far, the two exchanges we have don't overlap in sub web property ranges. To save space, we can use exchange ids instead of web_property names.

Commented [97]: Note that you can use the exchange id instead of the web property - these are mapped 1:1 for Awbid, but exchange id takes negligible space (fits into a byte).

Commented [98]: good suggestion. updated.

pushing data to production

There are generally two approaches to pushd live data to production, GDP or Supermessenger. GDP pushes files out multiple times during the day but does not work on the weekends; while supermessenger can do on demand live updates. Since our requirement is fully ramp up a domain in 5-7 days time, it's not super critical to use live updates, periodical snapshots is enough in this case.

Commented [99]: periodical snapshot probably is enough in this case

Commented [100]: noted, thanks.

However, XBFE has not set up GDP, we need to set up GDP from scratch, including modifying binary and change borg file, etc. This process can take several weeks. In addition, XBFE/XBid is trying to unify ways to push data to production. GDP has proven unreliable on XBid and they are moving away from it.

On the other hand, supermessenger is the SRE endorsed way of pushing data to production. There are already supermessenger channels exist in XBid. It's also recommended that we use a shared MLT instance instead of loading the whitelist in each XBFE task. Luckily, XBid already has an MLT instance that we can reuse, so no extra work is needed for setting up.

The drawback to this approach is that MLT lookup is an RPC call, which can adds up to 2 ms latency to the flow. Typical AWBid deadline is around 100 ms, we currently have 90 percentile latency of ~63 ms as of 9/3/2012. So we can take this RPC latency hit. Furthermore, we may be able to parallelize this RPC with other calls within XBFE to further minimize the latency impact.

Commented [101]: when this becomes an issue, we have an option to switch to local mlt. The switch will be straight forward and involves only slight code change inside of XBFE. channels/producers all stay same.

Commented [102]: It probably is not an issue now. but thanks for the suggestion, will definitely keep this in mind.

MLT lookup client

Either use MessengerLookupTable or BCMixerMultiLookupClient.
TODO (joanyuan): fill in more details and decide which one to use.

Rampup Algorithms

1) use a single threshold instead (average adx spam rate) of tiered approach
2) use the max of clean_impression_rate, clean_click_rate as probability to throttle traffic instead of multiplying those two numbers.

Project Information

This design is part of the AWBid project.

Caveats

List reasons why simpler approaches don't work. Mention other things to watch out for (if any).

Latency

As mentioned above, using an MLT look up in production incurs an extra latency up to 2 ms (co-locating in the same DC as the XBFE).

Scalability

This automated approach should scale well in terms of ramping up/down new domains.

Since whitelist is loaded into the MLT server, it does not incur extra memory for each XBFE task. As discussed previously, the whitelist file most likely is well below 50 Meg in the near future (till end of 2014).

Redundancy & Reliability

The offline program that generates the whitelist and the producer program that update the supermessgener channel should be reasonably reliable. We might not ramp up new publishers or not ramp down spammers quickly enough if they fail. At any rate, it should not be much worse than the manual blacklist process we currently have.

We don’t add new backends on the serving side, since MLT is already in place. Current supermessgener channel refresh and garbage collection policy suit our needs.

Security Considerations

Sketch your threat model and describe the system's security mechanisms. Be sure to describe any attack surfaces, any known vulnerabilities or points of failures, as well as any potentially insecure dependencies (e.g., does the application make any assumptions about access control lists in routers). If somehow your application doesn't have security considerations, explicitly state so (and why).

Additionally, projects must fill out a short, multiple-choice questionnaire at http://go/secq and paste the survey link here. Once you've done that, request a security review at http://go/securitydesignreview and include the bug in this doc. You should request a design review early in the process, ideally prior to significant implementation.

no new threat interface introduced. We have full control of the offline program to generate the whitelist.

Privacy

Redacted - Privilege

Testing Plan

What are the sub-units of your system that will be independently testable? Tests must be approved by the code reviewer, and must follow the guidelines in the unittesting document as far as possible. See What Is A Design Document for criteria for whole system tests on servers.
If there are changes envisaged in your future work, would your tests verify the base functionality? If some of your tests cannot be easily automated, how will you document the needed special procedures?

Work Estimates

Estimates of how long each phase will take (please be detailed; subtask granularity should be roughly one week)

- sstable generation: 1-2 weeks (parallel with below)
- supermessenger producer end: 1 week
- supermessenger set up/consumer end: 1 week
- setting up supermessgener channel tests/canary/monitoring: 1-2 weeks
- setting up test framework: 1-2 weeks

Code Location

borgcron job to generate whitelist sstable and copy to super messenger channels

whitelist sstable location known to supermessenger (/cns/[yh|ic|pa|ob]-d/home/contentads-awbid/whitelist/..)

Current Status

Launched on 12/5/2013.

Launch doc is here

Launch ticket

Future Work

- Fine tune ramp up plan, set up experiment to compare different ramp up plan
- Adding capability to 'learn' from existing info we learned about publishers, e.g. merge with a list of currently banned publishers from adx, or even from xbid

Potential Patents

Redacted - Privilege

Document History

List important milestones such as major revisions, reviews.